(51) International Patent Classification[7]: H04L 1/18, H04J 3/02

(21) International Application Number: PCT/US02/05573

(22) International Filing Date: 25 February 2002 (25.02.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/271,124   24 February 2001 (24.02.2001)   US

(71) Applicants and
(72) Inventors: BHANOT, Gyan, V. [US/US]; 347 Dodds Lane, Princeton, NJ 08540 (US). BLUMRICH, Matthias, A. [US/US]; 76 Florida Hill Road, Ridgefield, CT 06877 (US). CHEN, Dong [CN/US]; 15 Scenic Drive, Apt. K, Croton-On-Hudson, NY 10520 (US). COTEUS, Paul, W. [US/US]; 2742 Quinlan Street, Yorktown Heights, NY 10598 (US). GARA, ALAN, G. [US/US]; 38 Marion Avenue, Mount Kisco, NY 10549 (US). GIAMPAPA, Mark, E. [US/US]; 140 North Broadway, Apt. GT-14, Irvington, NY 10533 (US). HEIDELBERGER, Philip [US/US]; 24 Lakeview Avenue West, Cortlandt Manor, NY 10567 (US). STEINMACHER-BUROW, Burkhard, D. [CA/US]; 28 Manchester Drive, MOunt Kisco, NY 10549 (US). TAKKEN, Todd, E. [US/US]; 118 Foxwood Circle, Mount Kisco, NY 10549 (US). VRANAS, Pavlos, M. [GR/US]; 30-A Bedford Avenue, Bedford Hills, NY 10507 (US).

(74) Agents: GROLZ, Edward, W. et al.; Scully, Scott, Murphy & Presser, 400 Garden City Plaza, Garden City, NY 11530 (US).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW),

(54) Title: CLASS NETWORK ROUTING
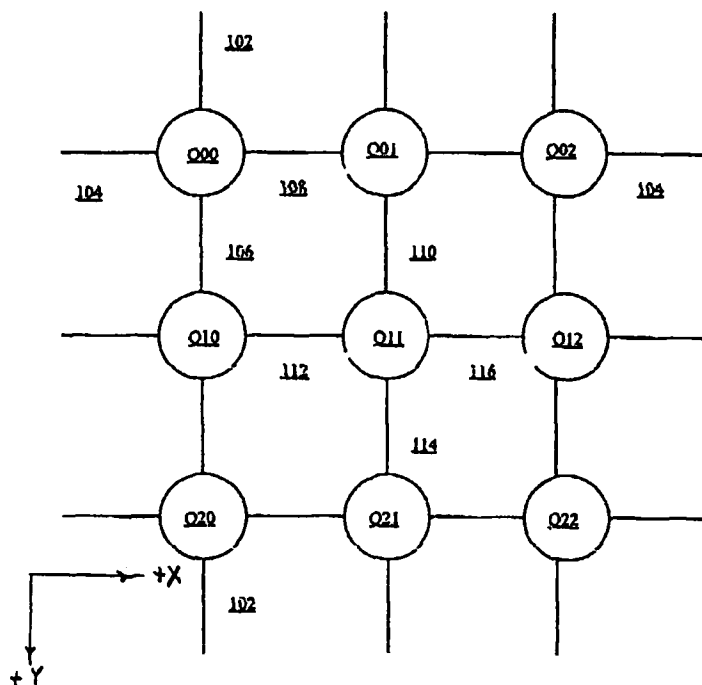
(57) Abstract: Class network routing is emplemented in a network such as a computer network comprising a plurality of parallel compute processors at nodes (Q00-Q22) thereof. Class network routing allows a compute processor to broadcast a message to a range (one or more) of other compute processors in the computer network, such as processors in a column or a row. Normally this type of operation requires a separate message to be sent to each processor. With class network routing pursuant to the invention, a single message is sufficient, which generally reduces the total number of messages in the network as well as the latency to do a broadcast. Class network routing is also applied to dense matrix inversion algorithms on distributed memory parallel supercomputers (Fig. 1) with hardware class function (multicast) capability. This is achieved by exploiting the fact that the communication patterns of dense matrix inversion can be served by hardware classe functions, which results in faster execution times.

WO 02/069550 A1

Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

---- *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

CLASS NETWORK ROUTING

CROSS-REFERENCE

The present invention claims the benefit of commonly-owned, co-pending United States Provisional Patent Application Serial Number 60/271,124 filed February 24, 2001 entitled MASSIVELY PARALLEL SUPERCOMPUTER, the whole contents and disclosure of which is expressly incorporated by reference herein as if fully set forth herein. This patent application is additionally related to the following commonly-owned, co-pending United States Patent Applications filed on even date herewith, the entire contents and disclosure of each of which is expressly incorporated by reference herein as if fully set forth herein. U.S. patent application Serial No. (YOR920020027US1, YOR920020044US1 (15270)), for "Class Networking Routing"; U.S. patent application Serial No. (YOR920020028US1 (15271)), for "A Global Tree Network for Computing Structures"; U.S. patent application Serial No. (YOR920020029US1 (15272)), for 'Global Interrupt and Barrier Networks"; U.S. patent application Serial No. (YOR920020030US1 (15273)), for 'Optimized Scalable Network Switch"; U.S. patent application Serial No. (YOR920020031US1, YOR920020032US1 (15258)), for "Arithmetic Functions in Torus and Tree Networks'; U.S. patent application Serial No. (YOR920020033US1, YOR920020034US1 (15259)), for 'Data Capture Technique for High Speed Signaling"; U.S. patent application Serial No. (YOR920020035US1 (15260)), for 'Managing Coherence Via Put/Get Windows'; U.S. patent application Serial No. (YOR920020036US1, YOR920020037US1 (15261)), for "Low Latency Memory Access And Synchronization"; U.S. patent application Serial No. (YOR920020038US1 (15276)), for 'Twin-Tailed Fail-Over for Fileservers Maintaining Full Performance in the Presence of Failure"; U.S. patent application Serial No. (YOR920020039US1 (15277)), for "Fault Isolation Through No-Overhead Link Level Checksums'; U.S. patent application Serial No. (YOR920020040US1 (15278)), for "Ethernet Addressing Via Physical Location for Massively Parallel Systems"; U.S. patent application Serial No. (YOR920020041US1 (15274)), for "Fault Tolerance in a Supercomputer Through Dynamic Repartitioning"; U.S. patent application Serial No. (YOR920020042US1

(15279)), for "Checkpointing Filesystem"; U.S. patent application Serial No.
(YOR920020043US1 (15262)), for "Efficient Implementation of Multidimensional
Fast Fourier Transform on a Distributed-Memory Parallel Multi-Node Computer";
U.S. patent application Serial No. (YOR9-20010211US2 (15275)), for "A Novel

5   Massively Parallel Supercomputer"; and U.S. patent application Serial No.
(YOR920020045US1 (15263)), for "Smart Fan Modules and System".

BACKGROUND OF THE INVENTION

10   1.Field of the Invention

The present invention relates generally to a class network routing, and more
particularly pertains to class network routing which implements class routing in a
network such as a computer network comprising a plurality of parallel compute

15   processors at nodes thereof, and which allows a compute processor to broadcast a
message to one or more other compute processors in the computer network, such as
processors in a column or a row. Normally this type of operation requires a separate
message to be sent to each processor. With class network routing pursuant to the
invention, a single message is sufficient, which generally reduces the total number

20   of messages in the network as well as the latency to do a multicast.

The present invention relates to the field of message-passing data networks, for
example, a network as used in a distributed-memory message-passing, parallel
computer, as applied for example to computation in the field of life sciences.

25
The present invention also uses the class function on a torus computer network to do
dense matrix calculations. By using the hardware implemented class function on the
torus computer network it is possible to do high performance dense matrix
calculations.

30

The present invention also relates to the field of distributed-memory, message-
passing parallel computer design and system software, as applied for example to
computation in the field of life sciences. More specifically it relates to the field of

high performance linear algebra software for distributed memory parallel supercomputers.

2. Discussion of the Prior Art

5   A large class of important computations can be performed by massively parallel computer systems. Such systems consist of many compute nodes, each of which typically consist of one or more CPUs, memory, and one or more network interfaces to connect it with other nodes.

The computer described in related U.S. provisional application Serial No.

10   60/271,124, filed February 24, 2001, for A Massively Parallel Supercomputer, leverages system-on-a-chip (SOC) technology to create a scalable cost-efficient computing system with high throughput. SOC technology has made it feasible to build an entire multiprocessor node on a single chip using libraries of embedded components, including CPU cores with integrated, first-level caches. Such

15   packaging greatly reduces the components count of a node, allowing for the creation of a reliable, large-scale machine.

A message-passing data network serves to pass messages between nodes of a network, each of which can perform local operations independently of other nodes.

20   Nodes can act in concert by passing messages between them over the network. An example of such a network is a distributed-memory parallel computer wherein each of its nodes has one or more processors that operate on local memory. An application using multiple nodes of such a computer coordinates the actions of the multiple nodes by passing messages between them. The words switch and router are

25   used interchangeably throughout this specification.

A message-passing data network consists of switches and links, wherein a link merely passes data between two switches. A switch routes incoming data from a node or link to another node or link. A switch may be connected to an arbitrary

30   number of nodes and links. Depending on their location in the network, a message between two nodes may need to traverse several switches and links.

Prior art networks efficiently support some types of message passing, but not all types. For example, some networks efficiently support unicast message passing to a

single receiving node, but not multicast message passing to an arbitrary number of receiving nodes. Efficient support of multicast message passing is required in various situations, such as numerical algorithms executed on a distributed-memory parallel computer, which is a requirement in the applications disclosed herein for

5   dense matrice inversion using class functions.

Many user applications need to invert very large N by N (NxN) dense matrices, where N is greater than several thousand. Dense matrices are matrices that have most of their entries being non-zero. Typically, inversion of such matrices can only

10   be done using large distributed memory parallel supercomputers. Algorithms that perform dense matrix inversions are well known and can be generalized for use in distributed memory parallel supercomputers. In that case a large amount of inter-processor communication is required. This can slow down the application considerably..

15

SUMMARY OF THE INVENTION

Accordingly, it is a primary object of the present invention to provide class network routing which implements class routing in a network which allows a compute processor to broadcast a message to a range of processors, such as processors in a

20   column or a row. Normally this type of operation requires a separate message to be sent to each processor. With class routing pursuant to the present invention, a single message is sufficient, which generally reduces the total number of messages in the network as well as the latency to do a broadcast. The class network routing enhances a network such that it more efficiently supports some additional types of

25   message passing.

Class routing enhances a network to more efficiently support additional types of message passing. As usual, a message is divided into one or more packets which pass atomically through the network. Class routing adds a class value to each packet.

30   At each switch, the class value is used as an index to one or more tables, whose stored values determine the actions performed by the switch on the packet. An index-based table-lookup is fast and efficient, as required for maximal throughput and minimal latency across a switch.

Class routing can be summarized as an efficient encoding and decoding of information needed by a switch to act on a packet, to enable the network to provide certain types of message passing. The information is encoded in the class value of the packet and in the tables of the switches. The information is decoded by using the

5   class value of a packet as an index to the tables.

A network without class routing is referred to as a basic network. With class routing, it is an enhanced network. With the appropriate entries in the class tables of all the switches, one or more classes of the enhanced network can provide the message-

10   passing types of the basic network. Moreover, since using the class value of a packet as an index to a table is fast, the message-passing types of the basic network are not appreciably slowed down by the enhancement when compared with the basic network.

15   Other entries in the class tables can provide message-passing types beyond those of the basic network. For example, the unicast message passing of a basic network can be enhanced by class routing to path-based multidrop message passing for multiphase multicasting.

20   In the classes described above, the enhanced network provides the message-passing types of the basic network, either unmodified or enhanced. In addition, some classes of the enhanced network could override the basic network. For example, overriding classes can provide multidestination message passing for single-phase multicasting. If class routing provides the only message-passing types, then no underlying basic

25   network is required.

The present invention makes dense matrix inversion algorithms on distributed memory parallel supercomputers with hardware class function capability perform faster. A hardware class function is a particular use of class routing. This is

30   achieved by exploiting the fact that the communication patterns of dense matrix inversion can be served by hardware class functions. This results in faster execution times.

If the parallel supercomputer possesses class function capability at the hardware

level, then the particular communication patterns of dense matrix inversion can be

exploited by using class functions in order to minimize the communication delay.

For example, provisional application Serial No. 60/271,124 describes a computer

5   with function capability at the hardware level.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing objects and advantages of the present invention for a class network

routing may be more readily understood by one skilled in the art with reference

10   being had to the following detailed description of several embodiments thereof,

taken in conjunction with the accompanying drawings wherein like elements are

designated by identical reference numerals throughout the several views, and in

which:

Figure 1 illustrates an exemplary distributed-memory parallel supercomputer that

15   includes 9 nodes interconnected via a multidimensional grid utilizing a 2-

dimensional 3x3 Torus network according to the present invention;

Figure 2 illustrates in more detail an exemplary node Q00 of the nine nodes of the

distributed-memory parallel supercomputer of Figure 1;

20

Figure 3 illustrates an exemplary single phase multicast from node Q00 to the other

8 nodes of the distributed-memory parallel supercomputer illustrated in Figure 1.

Figure 4 illustrates a 4 x 4 grid of processors wherein each processor is labeled by its

25   row, column numerals.

DETAILED DESCRIPTION OF THE INVENTION

The distributed-memory parallel supercomputer described in U.S. provisional

application Serial No. 60/271,124 comprises a plurality of nodes. Each of the nodes

30   includes at least one processor, which operates on a local memory. The nodes are

interconnected as a multidimensional grid and they communicate via grid links.

Without losing generality and in order to make the description of this invention

easily understandable to one skilled in the art, the multidimensional node grid will

6

be described as an exemplary 2-dimensional grid or an exemplary 3-dimensional grid. The 3-dimensional grid is implemented by a Torus-based architecture. Notwithstanding the fact that only the 2-dimensional node grids or 3-dimensional node grids are described in the following description, it is contemplated within the

5   scope of the present invention that grids of other dimensions may easily be provided based on the teachings of the present invention. An example of 3 dimensions is the 3-dimensional grid implemented on the Torus-based architecture described in provisional application Serial No. 60/271,124.

10   Figure 1 is an exemplary illustration of a distributed-memory parallel supercomputer that includes 9 nodes interconnected via a multidimensional grid utilizing a 2-dimensional 3x3 Torus network 100. It is noted that the number of nodes is in exemplary fashion limited to 9 nodes for brevity and clarity, and that the number of nodes may significantly vary depending on a particular architectural requirements

15   for the distributed-memory parallel supercomputer. Figure 1 depicts 9 nodes labeled as Q00 – Q22, a pair of which is interconnected by a grid link. In total, the 9-node Torus network 100 is interconnected by 18 grid links, where each node is directly interconnected to four other nodes in the Torus network 100 via a respective grid link.   It is noted that unlike a mesh, the exemplary 2-dimensional Torus network

20   100 includes no edge nodes. For example, node Q00 is interconnected to node Q20 via grid link 102; to node Q02 via grid link 104; to node Q10 via grid link 106; and finally to node Q01 via grid link 108. As another example, Node Q11 is interconnected to Node Q01 via grid link 110; to node Q10 via grid link 112; to node Q21 via grid link 114 and finally to Node Q12 via grid link 116. Other nodes

25   are interconnected in a similar fashion.

Data communicated between nodes is transported on the network in one or more packets. For any given communication, more than one packet is needed if the amount of data exceeds the packet-size supported by the network. A packet consists

30   of a packet header followed by the data carried by the packet. The packet header contains information required by the torus network to transport the packet from the source node of the packet to the destination node. In a distributed-memory parallel supercomputer, that is implemented by the assignee of the present patent application,

each node on the network is identified by a logical address and the packet header includes a destination address so that the packet is automatically routed to a node on the network as identified by a destination.

5   Figure 2 is an exemplary illustration of node Q00 of the distributed-memory parallel supercomputer of Figure 1. The node is similar to that in provisional application Serial No. 60/271,124. The node contains one processor which operates on local memory. The node contains a router which sends and receives packets on the grid links 102,104,106,108 connecting the node Q00 to its neighboring nodes

10 Q20,Q02,Q10,Q01, respectively, as illustrated in Figure 1. The node contains a reception buffer. If the router receives a packet destined for the local processor, the packet is placed into the reception buffer, from which the packet can be received by the processor. Depending on the application and the packet, the processor may write the contents of the packet into memory. The node contains an injection buffers

15 which operates in a first-in, first-out (FIFO) manner. If the CPU places a packet into an injection FIFO, once the packet reaches the head of the FIFO, the packet is removed from the FIFO by the router and the router places the packet onto a grid link toward the destination node of the packet.

20 The routing implemented by the router has several simultaneous characteristics. The characteristics are some of those described in provisional application Serial No. 60/271,124. The routing is a virtual cut-through routing. Thus if an incoming packet on one of the grid links is not destined for the processor, then the packet is forwarded by the router onto one of the outgoing links. This forwarding is

25 performed by the router without the involvement of the processor. The routing is a shortest-path routing. For example, a packet sent by node Q00 to node Q02 will travel over the grid link 104. Any other path would be longer. For another example, a packet sent by node Q00 to node Q11 will travel over the grid links 106 and 112 or over the grid links 108 and 110. The routing is an adaptive routing. There may be a

30 choice of grid links by which a packet can leave a node. In the previous example, the packet could leave the node Q00 via the grid link 106 or 108. For a packet leaving a node, adaptive routing allows the router to choose the less busy outgoing link for a packet or to choose the outgoing link based on some other criteria. Adaptive routing

is not just performed at the source node of a packet; adaptive routing also is performed at each intermediate node that a packet may cut through on the packet's way to the packet's destination node.

5   Class routing can be used to achieve a wide variety of types of message passing. Some of these types are described in the following examples which describe many details of class routing.

Example 1. Path-based multidrop message passing:

10   The network of a distributed-memory parallel computer is an example of a message-passing data network. Each node of such a computer has one or more processors that operate on their local memory. An application using multiple nodes of such a computer coordinates their actions by passing messages between them. An example of such a computer is described in provisional application Serial No. 60/271,124 for

15   A Massively Parallel Supercomputer. In that computer, each single node is paired with a single switch of the network. In that computer, the switches are connected to each other as a three dimensional (3D) torus. Thus in that computer, each switch is linked to six other switches. These links are to a switch in the positive direction and to a switch in the negative direction in each of the three dimensions. Each switch is

20   identified by its (x, y, z) logical address on the 3-dimensional torus. By contrast, in a computer using a 2-dimensional torus, each switch is identified by its (x, y) logical address. In Figure 1, the positive X direction is towards the right, and the positive Y direction is towards the bottom. In Figure 1, node Q00 has the logical address (0,0), node Q01 has logical address (0,1) and so on. Since each node is paired with a single

25   switch, a node has the address of its switch. By including a field for such a logical address in the packet header, the packet can efficiently and conveniently identify its destination node. Without class routing, the basic network only provides unicast message passing. If a switch is the destination of an incoming packet, then the packet is given to the local node. Otherwise, the packet is put onto a link towards to

30   the destination node.
       The following is an example using class routing to implement multidrop message passing. Each packet header has a field for a class value. This value is either 0 or 1. Each switch has a table used to determine if, in addition to the usual unicast routing

of the packet, a copy should be deposited at the local node. This assumes for the
original unicast message passing, that the processor is not involved when the router
forwards a packet from one of the incoming links to one of the outgoing links. This
assumption is satisfied by virtual cut-through routing, as implemented for example

5   in provisional application Serial No. 60/271,124. This assumes for the original
unicast message passing, that the processor is not involved when the router forwards
a packet from one of the incoming links to one of the outgoing links. This
assumption is satisfied by virtual cut-through routing, as implemented for example
in the provisional application Serial No. 60/271,124. For the class values [0,1], the

10   entries in this deposit table are [0,1] and demand that the packet is not deposited or
deposited, respectively. The table is illustrated below. The table only applies for a
packet at a node other than its destination node. A packet at its destination node is
deposited as in the usual unicast routing. Thus packets with class value 0 obey the
original unicast message passing. Packets with class value 1 perform path-based

15   multidrop message passing.

For a packet NOT destined for this node	
class value	deposit value
0	0
1	1

Path-based multidrop message passing can be used to implement multiphase
20   multicasting, as described for example in _D.K.Panda, S.Singal and P.Prabhakaran,
"Multidestination Message Passing Mechanism Conforming to Base Wormhole
Routing Scheme", PCRCW'94, LNCS 853, Springer-Verlag, pp.131-145, 1994_.
The first example described here is a two phase multicast from node (0,0) to the 9
nodes of the 3*3 torus illustrated in Figure 1. In the first phase, node (0,0) sends a

25   multidrop message with destination (0,2). In the second phase, each of the 3
recipients of the first phase simultaneously send a multidrop message. Node (0,0)
sends to (2,0); node (0,1) to (2,1) and node (0,2) to (2,2). At the end of the second
phase, all 9 nodes of the 2-dimensional torus have received the broadcast message.

30   The above assumes that in the original unicast message passing, when the source
node and destination node are in the same row, then the path of the packet is along

that row. A row is a group of nodes which have equal values for all but one of the dimensions of the torus or mesh. The assumption is guaranteed by shortest-path routing, as implemented for example in provisional application Serial No. 60/271,124. The above assumption also is guaranteed by the deterministic routing

5   implemented in the provisional application. By contrast, the above assumption is not satisfied by the congestion avoidance routing implemented elsewhere, which routes a packet via some random node.

The second example described here is a three phase multicast from node (0,0,0) to

10 the 125 nodes of the 5*5*5 cube with the corners (0,0,0) and (4,4,4). In the first phase, node (0,0,0) sends a multidrop message with destination (0,0,4). In the second phase, each of the 5 recipients of the first phase simultaneously send a multidrop message. Node (0,0,0) sends to (0,4,0); node (0,0,1) to (0,4,1) and so on. In the third phase, each of the 25 recipients of the second phase simultaneously send

15 a multidrop message. Node (0,0,0) sends to (4,0,0); node (0,0,1) to (4,0,1) and so on. At the end of the third phase, all 125 nodes of the cube have received the broadcast message.

The above example of a 3-phase multicast for the 3-dimensional cube is easily

20 generalized as follows. For a D-phase multicast from an origin node to all nodes of a D-dimensional cube wherein, in a first phase the origin node sends a multidrop message to all other nodes in one of the rows of the sending node, in a second phase each of the recipients of the first phase and the sender of the first phase simultaneously send a multidrop message to all other nodes in a row orthogonal to

25 the row of the first phase, in a third phase each of the recipients of the second phase and the senders of the second phase simultaneously send a multidrop message to all other nodes in a row orthogonal to the rows of the first and second phases, and so on in further phases such that all node of the cube receive the broadcast message after all the phases.

30

The implementation of path-based multidrop message passing using class routing offers advantages beyond existing implementations. For example, a particular existing implementation places the deposit value into the packet. In that implementation, every node on the path of the packet receives a copy of the packet.

In contrast, since each switch can have different entries in its deposit table, class routing allows a node with the deposit entries [0,0] to not receive a copy of a packet, even though the node is on the path of the multidrop packet. The table is illustrated below. For example, with several class values for multicasting, this allows for
5   several multicast groups, each with a different set of nodes.

For a packet NOT destined for this node	
class value	deposit value
0	0
1	0

10   **Example 2. Sending multidrop packets without knowing the recipients**

As described in Example 1, class routing allows a node with the deposit entries [0,0] for class values [0,1] to not receive a copy of a packet, even though the node is on the path of the multidrop packet. This information need not be known by the source node of the multidrop packet. In other words, class routing allows a node to source a
15   multidrop packet without knowing the recipients. However, in the network of Example 1, there is one exception, the destination node of the multidrop packet always will receive a copy of the packet. Thus if the destination node is to not receive a copy of the packet, this must be known by the source node such that it can use another destination.

20

For example, assume node (0,0) is the source of a multidrop packet originally destined for node (0,2). This may be a natural destination on a torus network of size 3*3, since nodes (0,0) through (0,2) are a complete row. If node (0,2) is to not receive a copy, then this must be known by node (0,0). If node (0,0) also knows that
25   node (0,1) is to receive a copy, then (0,1) can be used as the destination of the multidrop packet.

In order to solve the exception caused by the destination node, class routing allows each switch to have an additional table which determines if a copy of a packet
30   should be deposited at the destination node. To solve the above example, for node (0,2) the entries in this destination table are [1,0] for the class values [0,1]. The entry 0 for class 1, causes node (0,2) to not receive multidrop messages, even if it is the

destination. The entry 1 for class 0 allows node (0,2) to receive unicast messages as usual. The two tables are illustrated below.

For a packet destined for this node (0,2)	
class value	deposit value
0	1
1	0

5

For a packet NOT destined for this node (0,2)	
class value	deposit value
0	0
1	0

In the above example, node (0,2) is not a participant in the multicast with class value 1.

10   As a contrasting example, node (0,1) is a participant in the multicast with class value 1. The corresponding tables for node (0,1) are illustrated below.

For a packet destined for this node (0,1)	
class value	deposit value
0	1
1	1

For a packet NOT destined for this node (0,1)	
class value	deposit value
0	0
1	1

Example 3. Snooping:

15   Assume the network described above in Example 1, including its use of the class value 0 for the unicast messages of the basic network. A node can snoop, and acquire and store information on the unicast packets passing through its switch by using the entry 1 for class value 0 in the deposit table.

20   The table is illustrated below. In the example, the node is a participant in the multicast with class value 1. The table only applies for a packet at a node other than its destination node. In this example, a packet at its destination node is deposited as in the usual unicast routing.

For a packet NOT destined for this node	
class value	deposit value
0	1
1	1

An example use of such snooping is the investigation of the performance of the
network. Without snooping there may only be information on when the packet
entered the network at the source node and when it exited at the destination node.

5   With snooping, there can be information on when the packet passed through a node
on the path of the packet. Since there may be multiple valid paths between a pair of
nodes, snooping also can provide information on which particular path was used.
An example of a routing with multiple valid paths between a pair of nodes is
adaptive routing, as implemented for example in provisional application Serial No.

10   60/271,124.

Since each switch can have different entries in its deposit table, class routing allows
an arbitrary number of nodes to be snooping. If only a small fraction of nodes in the
network are snooping, then the measurements are a statistical sampling.

15

Snooping is an example use of class routing not specifically related to multicasting.

Example 4. Single Phase Multicast

In a single phase multicast, the message is injected once into the network by one of
the nodes. In contrast, in a multiphase multicast, the message is injected several

20   times into the network, perhaps by multiple nodes. For example, in the multiphase
multicast on the 3*3 node torus described above in Example 1, the message is
injected a total of 1+3=4 times by 3 different nodes. For example, in the multiphase
multicast on the 5*5*5 node torus described above in Example 1, the message is
injected a total of 1+5+25=31 times by 25 different nodes.

25

As well known, to provide single phase multicast, a switch must be able to duplicate
an incoming packet onto multiple outgoing links. In essence, the message
duplication performed by a node in multiphase multicasting is performed by a
switch in single phase multicasting.

14

The advantage offered by class routing for single phase multicasting is an efficient encoding and decoding of which of the outgoing switches do or do not receive a copy of a particular incoming packet. After a simple example describing the

5   encoding and decoding scheme offered by class routing, the scheme is compared to existing schemes.

The first example described here is the same multicast described in Example 1 from node (0,0) to the 9 nodes of the 3*3 torus illustrated in Figure 1. In Example 1 it is a

10   two phase multicast; here it is a single phase multicast. Here the pattern of messages across the network is chosen to be similar to that of Example 1.

Each packet header has a field for a class value. This value is either 0 or 1. Each switch has a table used to determine if the usual unicast routing of the packet is to be

15   performed or if the actions of single phase multicast routing are to be performed. Each entry in the table is a bit string of the format UDXY. If in a table entry U is 1, then the usual unicast routing is to be performed, otherwise not. If D is 1, then a copy of the packet is to be deposited at the local node, otherwise not. If X is 1, then a copy of the packet is to go out the positive X link, otherwise not. If Y is 1, then a

20   copy of the packet is to go out the positive Y link, otherwise not. The two links in the negative X and Y direction are irrelevant to the example and are ignored here for simplicity.

For class value 0, the entry in the table is 1000 on all nodes. Thus packets with class

25   value 0 obey the original unicast message passing. For class value 1, the entry in the table depends on the location of the switch in the network. The entry at each switch mimics the actions of the corresponding node in the multiphase multicast of Example 1.

30   At each node, the table is obeyed for all packets entering the node. If a packet has class value 0, then the UDXY=1000 identifies the packet as a unicast packet and only then is the destination of the packet examined.

For class value 1, switch (0,0) has the entry 0011. This assumes that the source node
of the multicast does not need another copy. The table for node (0,0) is illustrated
below.

For a packet at node (0,0)	
class value	UDXY value
0	1000
1	0011

5

Continuing with class value 1 for the other switches in the 3*3 torus, the switch
(0,1) has the entry 0111. The four switches (0,2), (1,0), (1,1), and (1,2) have the
entry 0101. The three switches (2,0), (2,1) and (2,2) have the entry 0100. The above
is a complete encoding of the information required for the example multicast using
10   class 1. In short, packets with class value 0 obey the original unicast message
passing. Packets originating from node (0,0) with class value 1 perform single phase
multicast routing.

The above UDXY values at each node for multicast from node (0,0) using class 1 is
15   illustrated in Figure 3. At each node, the circle is open if D=0, that is, if no copy of
the packet is to be deposited at the node. At each node, the circle is closed if D=1,
that is, if a copy of the packet is to be deposited at the node. At each node, there is
an arrow in the positive X direction, if X=1, that is, if a copy of the packet is to go
out the positive X link. At each node, there is an arrow in the positive Y direction, if
20   Y=1, that is, if a copy of the packet is to go out the positive Y link.

The second example described here is the same multicast described in Example 1
from node (0,0,0) to the 125 nodes of the 5*5*5 cube with the corners (0,0,0) and
(4,4,4). In Example 1 it is a three phase multicast; here it is a single phase multicast.
25   Here the pattern of messages across the network is chosen to be similar to that of
example 1.

Each packet header has a field for a class value. This value is either 0 or 1. Each
switch has a table used to determine if the usual unicast routing of the packet is to be

performed or if the actions of single phase multicast routing is to be performed. Each
entry in the table is a bit string of the format UDXYZ. If in a table entry U is 1, then
the usual·unicast routing is to be performed, otherwise not. If D is 1, then a copy of
the packet is to be deposited at the local node, otherwise not. If X is 1, then a copy

5   of the packet is to go out the positive X link, otherwise not. Similar for the bits Y
and Z. The three links in the negative X, Y and Z direction are irrelevant to the
example and are ignored here for simplicity.

For class value 0, the entry in the table is 10000 on all nodes. Thus packets with

10   class value 0 obey the original unicast message passing. For class value 1, the entry
in the table depends on the location of the switch in the network. The entry at each
switch mimics the actions of the corresponding node in the multiphase multicast of
Example 1.
For class value 1, switch (0,0,0) has the entry 00111. This assumes that the source

15   node of the multicast does not need another copy. The three switches (0,0,1) through
(0,0,3) have the entry 01111. Switch (0,0,4) has the entry 01110. The fifteen
switches in the x=0 plane with the corners (0,1,0), (0,1,4), (0,3,0) and (0,3,4) have
the entry 01110. The five switches (0,4,0) through (0,4,4) have the entry 01100. The
75 switches of the cube with the corners (1,0,0), (1,0,4), (3,0,0) and (3,0,4) have the

20   entry 01100. The 25 switches in the x=4 plane with the corners (4,0,0), (4,0,4),
(4,4,0) and (4,4,4) have the entry 01000. The above is a complete encoding of the
information required for the example multicast using class 1. In short, packets with
class value 0 obey the original unicast message passing. Packets originating from
node (0,0,0) with class value 1 perform single phase multicast routing.

25

·· In the above example of class routing for single phase multicasting, the UDXYZ bit
string determines onto which output ports a packet is to be duplicated. A similar bit
string is used in some existing implementations of single phase multicasting. An
example is described in _R.Sivaram, R.Kesavan, D.K.Panda, C.B.Stunkel,

30   "Architectural Support for Efficient Multicasting in Irregular Networks", IEEE
Trans. On Par. And Dist. Systems, Vol.12, No.5, May 2001_. Another example is
described in patent _US5333279: Self-timed mesh routing chip with data
broadcasting, D.Dunning_. In these existing implementations, a bit string similar to

the above UDXYZ for each switch is in the packet header. In contrast, in the above class routing implementation, the packet header merely contains the class value which is used at each switch to look up in a table the UDXYZ entry.

5   The above class routing implementation of single-phase multicasting is in some ways less general than these existing implementations, but the class routing is in some ways more efficient. For example, in the packet header, a field for a class value is much smaller than a field for a bit string for each switch. In the above example, the class value is 0 or 1 and thus can be stored in a one-bit field in the

10   header. In contrast, the above UDXYZ bit string would require a five-bit field in the header. Moreover, several fields for UDXYZ values would be required, since different switches have different values for UDXYZ. The smaller field in the header is more efficient since it consumes less of the physical bandwidth of the torus network, leaving more bandwidth for the application data. The smaller field also

15   allows for a smaller latency, since typically at a switch, the entire header must be received and checked for errors, before the packet can be forwarded.

Example 5. Single Phase Multicast from Any Node in the Network

The single phase multicast using class routing described in Example 4 allows a single node to be the source of the message. In the example on the 2-dimensional

20   3*3 torus, the source is the node (0,0). In the example on the 3-dimensional 5*5*5 torus, the source is the node (0,0,0). We'll name this a heterogeneous single phase multicast, since the class routing table has different values at different nodes. The table only is used for one of the input links.

25   Class routing also can be used to implement a single phase multicast where the source can be any node in the network. We'll name this a homogenous single phase multicast, since on a homogeneous network such as a torus the class routing tables have the same value on every node. On a single node, the class routing tables have different values on the different incoming links.

30   The first example described here is the same multicast described in Example 4 from node (0,0) to the 9 nodes of the 3*3 torus illustrated in Figure 1. In Example 4 it is a heterogeneous single phase multicast; here it is a homogenous single phase

multicast. Here the pattern of messages across the network is chosen to be similar to that of example 4.

In the heterogeneous single phase multicast of example 4, a packet arriving at a node
5   via any of the incoming links uses the same table to determine the actions to be performed by the switch on the packet based on the class value. As demonstrated in example 4, for the heterogeneous multicast, different nodes have different values in the table. By contrast, in the homogenous single phase multicast of this example, each incoming link on each switch has a table used to determine the actions to be
10   performed on an incoming packet. As demonstrated below, for the homogeneous multicast, different nodes have the same values in the tables.

Each packet header has a field for a class value. This value is either 0 or 1. Each incoming link on each switch has a table used to determine if the usual unicast
15   routing of the packet is to be performed or if the actions of single phase multicast routing is to be performed. Each entry in the table is a bit string of the format UDXY. If in a table entry U is 1, then the usual unicast routing is to be performed, otherwise not. If D is 1, then a copy of the packet is to be deposited at the local node, otherwise not. If X is 1 and the X-destination of the packet is not the X-
20   location of the node, then a copy of the packet is to go out the positive X link, otherwise not. If Y is 1 and the Y-destination of the packet is not the Y-location of the node, then a copy of the packet is to go out the positive Y link, otherwise not. For each node, the two outgoing links in the negative X and Y directions are irrelevant to the example and are ignored here for simplicity. For each node, the two
25   incoming links in the negative X and Y directions are irrelevant to the example and are ignored here for simplicity.

As described above, the X-destination and the Y-destination of the packet are determined in order to determine the actions performed on the packet. Thus for node
30   (0,0) to broadcast to all other 8 nodes of the 3*3 torus, the packet must have the destination (3,3).

In general for a broadcast in this example, the destination of the packet is the furthest node in the positive X and positive Y direction from the source of the broadcast. For example, for node (1,0) to broadcast to all other 8 nodes of the 3*3 torus, the packet must have the destination (0,2).

5

For class value 0, the entry in the table is 1000 on all tables on all nodes. Thus packets with class value 0 obey the original unicast message passing. For class value 1, the entry in the table depends on which incoming link the packet arrived on. The tables are illustrated below. The entry for each incoming link are such that the

10   resulting homogeneous multicast mimics the heterogeneous multicast of Example 4.

For a packet incoming on the link from the negative x direction	
class value	UDXY value
0	1000
1	0111

For a packet incoming on the link from the negative y direction	
class value	UDXY value
0	1000
1	0011

15

The above is a complete encoding of the information required for the example multicast using class 1. In short, packets with class value 0 obey the original unicast message passing. Packets with class value 1 perform a homogeneous single phase multicast routing.

20

Given the above 2-dimensional torus example, the technique is easily extended to other networks. Class 1 in the above example can be considered to provide multicasting in the positive X and positive Y quadrant of a mesh. Three additional similar classes 2, 3 and 4 could provide multicasting in the other three quadrants:

negative X and positive Y; positive X and negative Y; as well as negative X and negative Y. These four classes allow any node in the mesh to use four multicasts to effectively broadcast a packet to all other nodes in the mesh. Using the same broadcast technique on the torus would be twice as fast as the single class technique

5   described above. It is twice as fast since the distance between the source node and the destination nodes is halved. This technique is feasible since any node on a torus can be treated as a node in the middle of a mesh.

The above technique is easily generalized to a mesh or torus of D dimensions. On a
10   D dimensional mesh or torus, 2^D classes allow any node in the mesh or torus to use 2^D multicasts to effectively broadcast a packet to all other nodes in the mesh or torus. On the torus, the alternative single broadcast to all the nodes will require twice as long to complete as the 2^D multicasts on the torus since the distance between the source node and the furthest destination is double for the single

15   broadcast.

Enhancements and Alternatives to Class Tables

Instead of or in addition to using tables on the switch, the class value and perhaps other characteristics of the packet can be input to an algorithm. If table entries are
20   the same for all class values, then it might be better to use a algorithm If a switch needs to decide between conflicting actions demanded by tables, as which can be programmed with the relative priorities of different tables.

<u>Using Class-based Multicasting to Create other Classes</u>
25   In Example 5, class value 0 is used for the usual unicast, while class value 1 can be used to broadcast to all nodes in the torus. Having established a broadcast mechanism, it can be used to broadcast any data. For example, this data could be the class table entries for other classes. For example, Example 5 identified a need for the additional classes 2,3 and 4. Once multicasting on class 1 is established by
30   whatever means, class 1 can be used to create classes 2,3 and 4. In general, once communication on a particular class value or values is established, that communication can be used to establish communication on other class values.

21

Example 7. Dense Matrix Calculation using Class Function

The present invention also uses the class function on a torus computer network to do dense matrix calculations. By using the hardware implemented class function on the torus computer network it is possible to do high performance dense matrix

5   calculations.

Class function is the name used in this example for multicasting based on class network routing. Often, the multicast is to other nodes in the same row. So often it is sufficient for class routing to implement a single phase of path-based multidrop

10   message passing, which is described in Example 1. When the multicast is not to a row, it is to a plane, cube or other higher dimension subset of the torus or mesh. In this case, optimal performance demands that class routing implement a more sophisticated multicast, such as the single phase multicast described in Example 5.

15   The present invention makes dense matrix inversion algorithms on distributed memory parallel supercomputers with hardware class function capability perform faster. This is achieved by exploiting the fact that the communication patterns of dense matrix inversion can be served by hardware class functions. This results in faster execution times.

20   The algorithms as discussed herein are well known in the art, and are discussed, for example, in NUMERICAL RECIPES IN FORTRAN, THE ART OF SCIENTIFIC COMPUTING, Second Edition, by William H. Press, et al., particularly at page 27 et. seq.

25   Figure 4 illustrates a 4 x 4 grid of processors wherein each processor is labeled by its row, column numerals. For example the processor in row 2 column 3 is p (2,3). The column i and row i are also shown (shaded areas) as well as the directions that the column/row has to be sent via the class function.

30   One can invert a dense linear matrix using standard algorithms such as Gauss-Jordan elimination as well as other methods. In general the I/O required is of a special one-to-many variety that is well suited to the communication functionality of a parallel supercomputer with hardware class function capability. One can utilize the class functionality to multicast data to an entire row or surface of the machine.

Some of the terms used in the description of this invention are explained below:

The Gauss-Jordan algorithm:

5   The kernel of the Gauss-Jordan algorithm without pivoting is given below. Initially
b is an identity matrix and a is the matrix whose inverse is being computed.

```
        do i=1,N
          do j=i,N
10          do k=1,N; (k not equal to i)
            b(k,j) = b(k,j) - [a(k,i) / a(i,i)] * b(i,j)
            a(k,j) = a(k,j) - [a(k,i) / a(i,i)] * a(i,j)
              enddo
            enddo
15        enddo
```

Equation 1

Distributed memory parallel supercomputer:

20   Such a computer consists of many nodes. Each node has one or more processors
that operate on local memory. The nodes are typically connected as a d-dimensional
grid and they communicate via the grid links. If the grid is 2-dimensional with PxP
processors then an NxN matrix can be partitioned so that LxL pieces of it reside on
each node

25   (L = N / P). If the machine is not connected as a 2-dimensional grid the problem can
always be mapped onto it by appropriately "folding" the matrix onto the grid.
Without loss of generality and in order to make the presentation of this invention
simple the processor grid will be assumed to be 2-dimensional.

30   *Hardware class functions:*
Class functions are a hardware implementation of multicast. Suppose that processor
p(1,1) (here the numerals indicate the position of the processor on the grid, also see
Figure 4) wants to send the same packet of data to processors p(1,2), p(1,3) and

p(1,4). Typically this is done by first sending the data to processor p(1,2). Once the
data arrives into p(1,2) software routines read it and store it in memory. Then p(1,2),
reads the data from memory and sends it to p(1,3) etc.. The problem with this is that
it takes a long time to fully receive the packet of data into memory and then resend

5   it. If the hardware was built so that the packet of data that arrived into p(1,2) was
simultaneously stored into the p(1,2) memory and immediately sent to p(1,3) then
the delay would be greatly reduced. The hardware function of p(1,1) sending a
packet of data to p(1,4) while that packet is deposited into the memory of the
intermediate processors that it goes through is called the hardware class function.

10

The invention:

This invention exploits the fact that the communication patterns of dense matrix
inversion (for example using the Gauss-Jordan method) can utilize class functions.

15   This can be seen from equation 1 that describes the Gauss-Jordan algorithm:

The a(i,i) are communicated via some other method, for example a global broadcast.
Then the right hand side of the equations for b(k,j) and a(k,j) involve elements that
have only one index different from (k,j) but not both (a(k,i), a(i,j) and b(i,j)). Class

20   function communication can be used to send such elements across the relevant
processors.
For example, in order to calculate b(k,j) for a given row k (1 < j < N) one needs
a(k,i) to be known for all processors that contain the row k. Therefore, one must
send a(k,i) along the row of processors that contain the matrix row k. This can be

25   done using the class functionality. As already discussed this results in large
reductions in total communication time.

This completes the description of the idea for this invention. The idea was described
for the Gauss-Jordan algorithm but it is not specific to it. For example this idea

30   applies to the "Gauss-Jordan with Pivoting", "Gaussian Elimination with Back
Substitution" and "LU Decomposition" algorithms.

An implementation of this idea (using the Gauss-Jordan algorithm) with all the
details is presented below as an example. In order to make the example easy to

understand the simplest implementation was chosen. More complex
implementations that result in communications involving larger data packets have
also been worked out. Depending on the size of the processor grid and the size of the
matrix larger packet sizes may be desirable since they further improve performance
5   by minimizing latency. However, this does not affect the premise of this idea.

An example algorithm:

The Gauss-Jordan algorithm is used to find the matrix inverse of a dense matrix of
10   size NxN uniformly spread out on a grid of PxP nodes. Therefore each node has an
LxL piece of the matrix in its memory (L=N/P). A hardware class function is used to
multicast data across rows and columns. For a visual picture of this algorithm please
refer to Figure 1 above.

15   For each $1 < i < N$
1) Using class functions send to the left and right the column i of a's ($a(k,i)$, $1 < k <$
N)

2) Scale the elements a, b of row i by $a(i,i)$
20

3) Using class functions send up and down the new row i of a's and b's ($a(i,j)$ and
$b(i,j)$, $1 < j < N$)

4) Now all processors have the necessary elements to do the standard Gauss-Jordan
25   step for column i. At the end of this column i is the same as column i of the identity
matrix.

Repeat

30   *End of examples:*

While several embodiments and variations of the present invention for class
networking routing are described in detail herein, it should be apparent that the

disclosure and teachings of the present invention will suggest many alternative designs to those skilled in the art.

CLAIMS

Having thus described our invention, what we claim as new and desire to secure by Letters Patent is:

1
1   1.   A method of class network routing in a network to allow a compute

2   processor in a network of compute processors located at nodes of the network to

3   multicast a message to a plurality of other compute processors in the network

4   comprising:

5   dividing a message into one or more message packets which pass

6   through the network;

7   adding a class value to a message packet;

8   at each switch in the network, using the class value as an index to at

9   least one table whose stored values, or as an input to an algorithm whose generated

10 values, determine actions performed by the switch on the message packet.

1   2.   The method of claim 1, including providing a class value determining a

2   switch action of path-based multidrop message passing for multiphase multicasting

3   of a message packet through the network, to determine if a local node should deposit

4   a copy of the message packet at the local node.

1   3.   The method of claim 2, including:

2   providing a class value to implement multidrop message passing;

3   providing each switch with a table to determine if a copy of the

4   message packet is to be deposited at the local node.

1   4.   The method of claim 1, including providing a class value determining a

2   switch action of multidestination message passing of a message packet to multiple

3   destination nodes in the network.

1   5.   The method of claim 1, wherein a switch duplicates an incoming packet onto

2   multiple outgoing links.

1 6.   The method of claim 5, including providing a class routing table with

2 different values on different incoming links.

1 7.   The method of claim 4, wherein the message packet is multicast to an entire

2 row or surface of the network.

1 8.   The method of claim 1, including:

2   performing dense matrix inversion algorithms on a network of

3 distributed memory parallel computers with hardware class function multicast

4 capability, wherein the hardware class function multicast capability simultaneously

5 stores into memory a message packet that arrives and immediately sends the

6 message packet to one or more other nodes while that message packet is being

7 stored into memory, such that the communication patterns of the dense matrix

8 inversion algorithms are served by the hardware class function multicast capability

9 to minimize communication delays.

1 9.   The method of claim 1, wherein the network comprises a network of

2 distributed-memory parallel computers;

3   providing each node of the computer network with one or more

4 processors that operate on local memory;

5   coordinating the actions of multiple nodes of the computer by using

6 class routing to pass messages between the multiple nodes.

1 10.   The method of claim 9, including:

2   pairing each node with a switch of the network;

3   connecting the switches to form a three dimensional torus wherein

4 each switch is linked to six other switches, the links are coupled to a switch in a

5 positive direction and also to a switch in a negative direction in each of the three

6 dimensions;

7   identifying each switch by an x, y, z logical address on the torus,

8 wherein each node has the address of its switch;

9   including a field value for the logical address in the packet header, to

10 enable the packet to identify a destination node.

1   11.   The method of claim 1, including using a D-phase multicast from an origin

2   node to all nodes of a D-dimensional cube wherein, in a first phase the origin node

3   sends a multidrop message to all other nodes in one of the rows of the sending node,

4   in a second phase each of the recipients of the first phase and the sender of the first

5   phase simultaneously send a multidrop message to all other nodes in a row

6   orthogonal to the row of the first phase, in a third phase each of the recipients of the

7   second phase and the senders of the second phase simultaneously send a multidrop

8   message to all other nodes in a row orthogonal to the rows of the first and second

9   phases, and so on in further phases such that all node of the cube receive the

10   broadcast message after all the phases.

1   12.   The method of claim 1, including providing each switch with a table with

2   associated class values which determine if a copy of a message packet is to be

3   deposited at a destination node.

1   13.   The method of claim 1, for a D-dimensional network, including providing

2   2^D class values for multicast in each of the 2^D directions to allow each node in

3   the network to use 2^D multicasts to effectively broadcast a packet to all other nodes

4   in the mesh.

1   14.   The method of claim 1, including providing a class value determining a

2   switch action of a unicast of a message packet through the network to a single

3   destination node.

1   15.   The method of claim 1, including providing a class value to enable a node to

2   acquire and store information packets passing through its switch to provide

3   information on the performance of the network.

1   16.   The method of claim 11, including providing class values to determine if a

2   copy of the message packet is to go out on an X link or not, and out on a Y link or

3   not, and out on a Z link or not and so on for the other links of the D dimensions.

1   17.   The method of claim 1, including providing different tables and providing

2   priorities for different tables to enable a switch to decide between conflicting actions

3   indicated by different tables.

1   18.   The method of claim 1, including using a class value as an input to an

2   algorithm for determining a switch action of the switch.

1   19.   The method of claim 1, including using class-based multicasting to create

2   other classes, such that the contents of a table for a particular class value is

3   determined by using another class value.
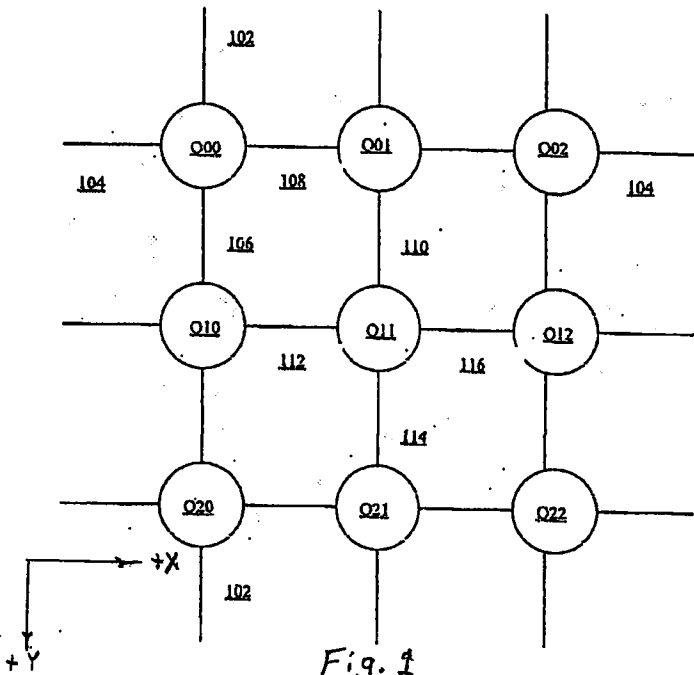
Fig. 1
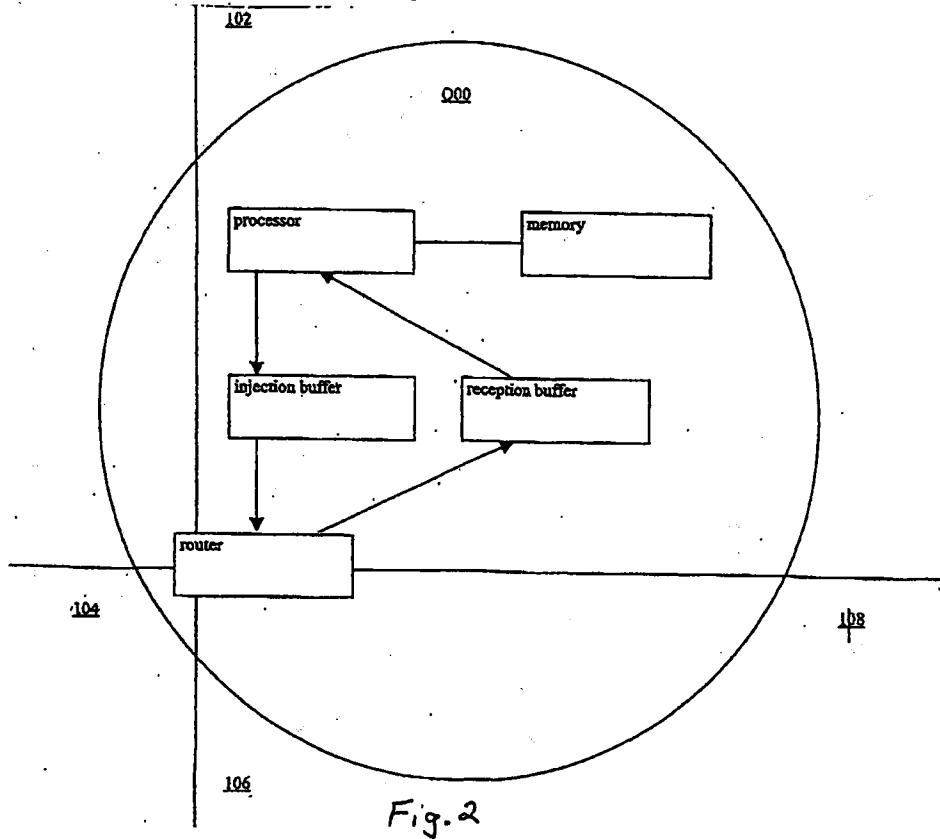
Fig. 2
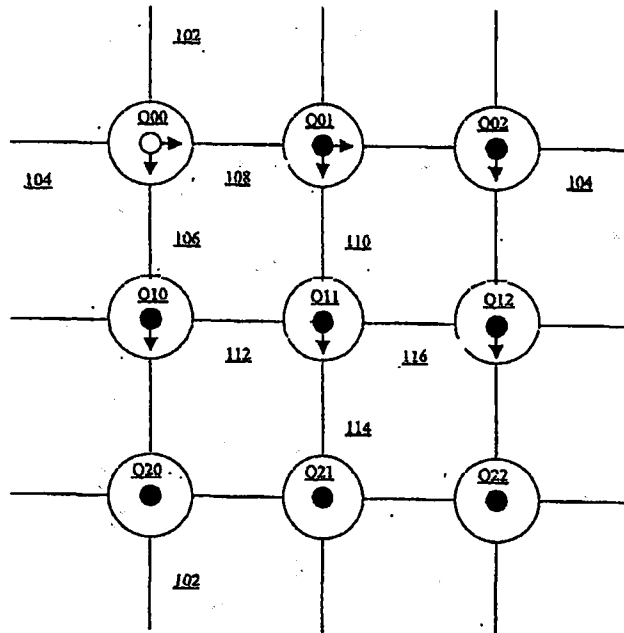
Fig. 3
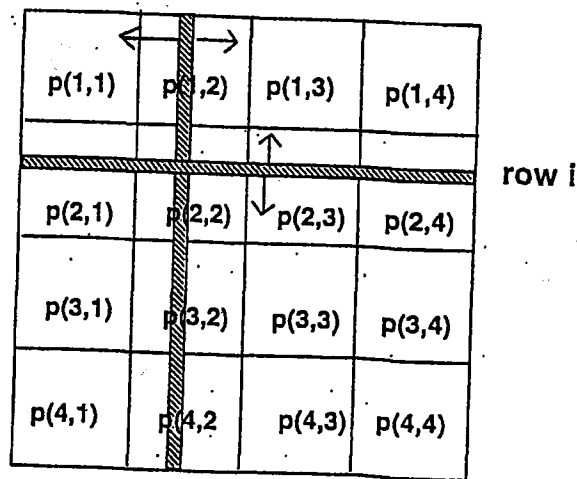
column i   Fig. 4

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US09/05573

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :H04L 1/18; H04J 3/02

US CL : 714/748; 370/94.3

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 714/748; 370/94.8

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,112,323 A (MEIZLIK et al) 29 August 2000, ALL	1-19
A	US 5,151,900 A (SNYDER et al) 29 September 1992, ALL	1-19

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

Date of the actual completion of the international search	Date of mailing of the international search report
12 JUNE 2002	10 JUL 2002

| Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231 | Authorized officer

ALEXANDER BOAKYE |
| Facsimile No.   (703) 305-3230 | Telephone No.   (703) 308-5 |

Form PCT/ISA/210 (second sheet) (July 1998)*